

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 06-314221

(43)Date of publication of application : 08.11.1994

(51)Int.Cl.

G06F 11/34
G06F 9/45

(21)Application number : 05-105213

(71)Applicant : MITSUBISHI ELECTRIC CORP

(22)Date of filing : 06.05.1993

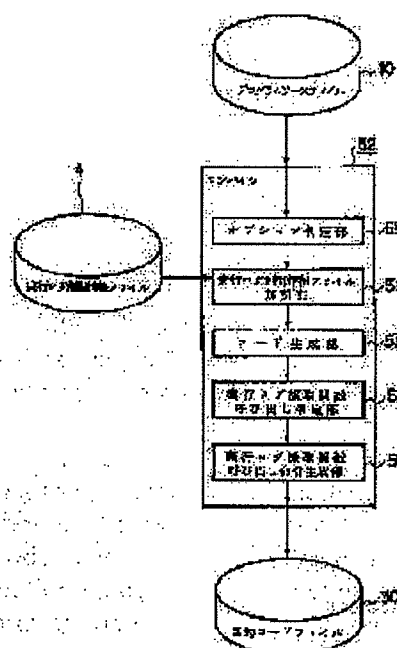
(72)Inventor : FUKUCHI YUSHI

(54) COMPILING SYSTEM

(57)Abstract:

PURPOSE: To provide a compiling system generating a target code file into which a code for calling a designated execution log extraction function is automatically inserted at an arbitrary part in a source program without correcting the source program.

CONSTITUTION: A compiler 52 using a compiling system is provided with an option judgement part 55 judging whether an execution log extraction option is designated for an option or not, an execution log control file analysis part 56 reading and analyzing execution log control information from an execution log control information file 4 designated for the execution log extraction option, an execution log extraction function call judgement part 58 judging the part into which the execution log extraction function of the source program is inserted based on execution log control information, and an execution log extraction function call instruction generation part 59 generating the code calling the execution log extraction function, and the target code file 30 is generated.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19)日本国特許庁 (J P)

(12) 公 開 特 許 公 報 (A)

(11)特許出願公開番号

特開平6-314221

(43)公開日 平成6年(1994)11月8日

(51)Int.Cl. ⁵	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 11/34	P	9290-5B		
9/45		9292-5B	G 0 6 F 9/ 44	3 2 2 A

審査請求 未請求 請求項の数3 OL (全 8 頁)

(21)出願番号 特願平5-105213

(22)出願日 平成5年(1993)5月6日

(71)出願人 000006013

三菱電機株式会社

東京都千代田区丸の内二丁目2番3号

(72)発明者 福地 雄史

神奈川県鎌倉市大船五丁目1番1号 三菱

電機株式会社情報電子研究所内

(74)代理人 弁理士 金山 敏彦 (外2名)

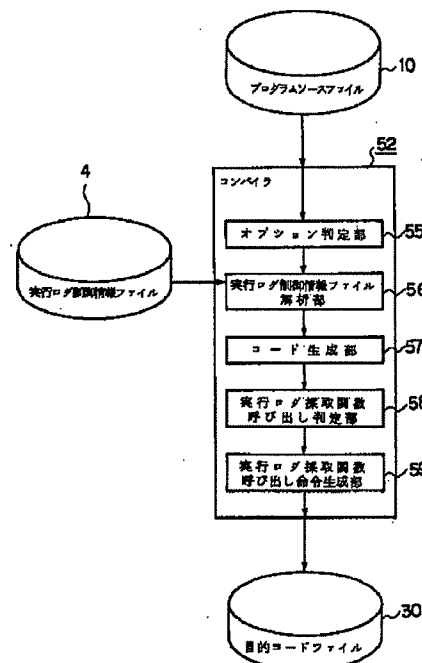
(54)【発明の名称】 コンパイル方式

(57)【要約】

【目的】 ソースプログラムを修正することなく、ソースプログラム中の任意の箇所指定実行ログ採取関数を呼び出すコードを自動的に挿入した目的コードファイルを生成するコンパイル方式を提供する。

【構成】 本発明におけるコンパイル方式を使用したコンパイラ52は、オプションに実行ログ採取オプションが指定されているかどうかを判定するオプション判定部52と、実行ログ採取オプションに指定された実行ログ制御情報ファイル4から実行ログ制御情報を読み込み解析する実行ログ制御ファイル解析部56と、実行ログ制御情報に基づき、ソースプログラムの実行ログ採取関数を挿入する箇所を判定する実行ログ採取関数呼び出し判定部58と、実行ログ採取関数を呼び出すコードを生成する実行ログ採取関数呼び出し命令生成部59と、から成り、目的コードファイル30を生成する。

本実施例におけるコンパイラの内部処理



【特許請求の範囲】

【請求項1】 ソースプログラムの実行におけるログ情報を採取するための目的コードを生成するコンパイル方式において、
コンパイラ起動時に指定されたオプションを解析し、実行ログ採取オプションが指定されているかどうかを判定するオプション判定手段と、
前記実行ログ採取オプションに指定された実行ログ制御情報ファイルから実行ログ制御情報を読み込み解析する実行ログ制御ファイル解析手段と、
前記実行ログ制御ファイル解析手段により解析された前記実行ログ制御情報に基づき、前記ソースプログラムの実行ログ採取関数を挿入する箇所を判定する実行ログ採取関数呼び出し判定手段と、
前記実行ログ採取関数呼び出し判定手段により指定された箇所に挿入する前記実行ログ採取関数を呼び出すコードを生成する実行ログ採取関数呼び出し命令生成手段と、
を有することを特徴とするコンパイル方式。

【請求項2】 ソースプログラムの実行におけるログ情報を採取するための目的コードを生成するコンパイル方式において、
コンパイラ起動時に指定されたオプションを解析し、実行ログを採取するかどうかを判定するオプション判定手段と、
前記オプションに指定された実行ログ制御情報に基づき、前記プログラムの実行ログ採取関数を挿入する箇所を判定する実行ログ採取関数呼び出し判定手段と、前記実行ログ採取関数呼び出し判定手段により指定された箇所に挿入する前記実行ログ採取関数を呼び出すコードを生成する実行ログ採取関数呼び出し命令生成手段と、
を有することを特徴とするコンパイル方式。

【請求項3】 請求項1記載のコンパイル方式において、
前記実行ログ制御情報には、1ないし複数の実行ログ制御情報ファイル名が含まれていることを特徴とするコンパイル方式。

【発明の詳細な説明】

【0001】

【産業上の利用分野】この発明は、コンパイル方式、特に、ソースプログラムを変更することなく実行ログ採取関数をプログラム内部の任意の位置に自動的に組み込み実行ログ情報を得るコンパイル方式に関する。

【0002】

【従来の技術】図3は、特開平4-287134号公報に記載されている従来のコンパイル方式による実行ログ採取方法のコンパイラのブロック図であり、同時に処理、データの流れも示した図である。図3において、1はコンパイルされるべきソースプログラムが格納されているプログラムソースファイル、2はコンパイラ、3は

コンパイラ2によるコンパイルの結果得られる目的コードファイルである。5はオプションが指定されているかどうかを判定するオプション判定部、7はプログラムソースファイル1をコード化するコード生成部、8はコード生成部7により生成されたコードが関数呼び出しのコードかどうかを判定する実行ログ採取関数呼び出し判定部、9は実行ログ採取関数呼び出し命令を生成する実行ログ採取関数呼び出し命令生成部である。

【0003】次に図3を用いて従来のコンパイラ2の内部処理の手順を説明する。

【0004】ユーザは、作成したソースプログラムの実行状況を調べるためにそのプログラムの実行状況をログ情報として採取して調査する手法をとる場合、実行ログ情報採取対象となる作成したプログラムソースファイル1に実行ログ採取オプションをオプションとして指定してコンパイルする。コンパイラ2は、オプション判定部5において、まず実行ログ採取オプションが指定されているかを判定する。その後、コード生成部7において、通常の構文解析処理を行い、最後にコード生成処理を行う。ここで、実行ログ採取オプションが指定されていれば、実行ログ採取関数呼び出し判定部8において、生成されたコードが実行ログ採取関数呼び出しのコードかどうかを判定する。実行ログ採取関数呼び出しのコードであれば、実行ログ採取関数呼び出し命令生成部9において、実行ログ採取関数呼び出し命令を生成して、目的コードファイル3を生成する。

【0005】

【発明が解決しようとする課題】従来の実行ログ情報採取機能を有するコンパイル方式は、実行ログ情報を得るためには、予めソースプログラムの中に実行ログ採取関数を呼び出すように指定する必要があるため、ソースプログラムの中の実行ログ採取の対象となる位置及びその位置で呼び出される実行ログ採取関数が限定されてしまうという問題があった。

【0006】また、ソースプログラムの中に実行ログ採取関数を指定する際、ユーザ自身によって必要な位置に必要な実行ログ採取関数の呼び出し文をソースプログラム中に追加あるいは変更を行わなくてはならず、その時間及び手間がかかるという問題があった。

【0007】更に、実行ログ情報採取終了後には再度ソースプログラムを修正してこの挿入された実行ログ採取関数を削除する作業が必要となり、この作業により新たなコーディングミスが発生してしまうなどの問題があった。

【0008】この発明は上記のような課題を解消するためになされたもので、ソースプログラムを修正することなく、ソースプログラム中の任意の箇所に指定実行ログ採取関数を呼び出すコードを自動的に挿入した目的コードファイルを生成するコンパイル方式を提供することを目的とする。

【0009】

【課題を解決するための手段】以上の目的を達成するために、本発明に係るコンパイル方式は、ソースプログラムの実行におけるログ情報を採取するための目的コードを生成するコンパイル方式において、コンパイラ起動時に指定されたオプションを解析し、実行ログ採取オプションが指定されているかどうかを判定するオプション判定手段と、前記実行ログ採取オプションに指定された実行ログ制御情報ファイルから実行ログ制御情報を読み込み解析する実行ログ制御ファイル解析手段と、前記実行ログ制御ファイル解析手段により解析された前記実行ログ制御情報に基づき、前記ソースプログラムの実行ログ採取関数を挿入する箇所を判定する実行ログ採取関数呼び出し判定手段と、前記実行ログ採取関数呼び出し判定手段により指定された箇所に挿入する前記実行ログ採取関数を呼び出すコードを生成する実行ログ採取関数呼び出し命令生成手段と、を有することを特徴とする。

【0010】また、他の発明によれば、ソースプログラムの実行におけるログ情報を採取するための目的コードを生成するコンパイル方式において、コンパイラ起動時に指定されたオプションを解析し、実行ログを採取するかどうかを判定するオプション判定手段と、前記オプションに指定された実行ログ制御情報に基づき、前記プログラムの実行ログ採取関数を挿入する箇所を判定する実行ログ採取関数呼び出し判定手段と、前記実行ログ採取関数呼び出し判定手段により指定された箇所に挿入する前記実行ログ採取関数を呼び出すコードを生成する実行ログ採取関数呼び出し命令生成手段と、を有することを特徴とする。

【0011】更に、他の発明によれば、上記発明におけるコンパイル方式において、前記実行ログ制御情報には、1ないし複数の実行ログ制御情報ファイル名が含まれていることを特徴とする。

【0012】

【作用】この発明におけるコンパイル方式によれば、オプション判定手段においてコンパイラ起動時に指定されたオプションを解析し、実行ログ採取オプションが指定されているかどうかを判定する。指定されていれば、実行ログ制御ファイル解析手段は、オプションに実行ログ採取オプションが指定されている実行ログ制御情報ファイルから実行ログ制御情報を読み込み、どの実行ログ採取関数をソースプログラムのどこに挿入するかを解析する。実行ログ採取関数呼び出し判定手段は、実行ログ制御情報に基づき、コード化されたソースプログラムのどの位置に実行ログ採取関数を挿入するかを判定する。実行ログ採取関数呼び出し命令生成手段は、実行ログ採取関数呼び出し判定手段により指定された箇所に挿入する実行ログ採取関数を呼び出すコードを生成する。このようにして、本発明におけるコンパイル方式を使用したコンパイラは目的コードファイルを生成する。

【0013】また、上記発明において、実行ログ採取オプションに直接実行ログ制御情報を指定することができる。

【0014】また、上記発明において、実行ログ制御情報ファイルに実行ログ制御情報を含む1ないし複数の実行ログ制御情報ファイル名を記述することができる。

【0015】以上のように、本発明によれば、ソースプログラムを修正することなく、任意の実行ログ採取関数をソースプログラム任意の位置に挿入することができる。

【0016】

【実施例】

実施例1. 以下、本発明における一実施例を図を用いて説明する。図1は、本実施例におけるコンパイル方式を使用したコンパイラのブロック図であり、同時に処理、データの流れも示した図である。

【0017】図1において、10はプログラムソースファイル、52はプログラムソースファイル10をコンパイルするコンパイラ、30はコンパイラ52によって生成された目的コードファイル、4は後述する実行ログ制御情報が記述されている実行ログ制御情報ファイルである。

【0018】コンパイラ52は、オプション判定部55、実行ログ制御ファイル解析部56、コード生成部57、実行ログ採取関数呼び出し判定部58及び実行ログ採取関数呼び出し命令生成部59から構成される。オプション判定部55は、コンパイラ52の起動時に指定されたオプションを解析し、実行ログ採取オプションが指定されているかどうかを判定する。実行ログ制御ファイル解析部56は、実行ログ採取オプションに指定された実行ログ制御情報ファイル4から実行ログ制御情報を読み込み、解析した後、実行ログ採取に関するデータをコンパイラ52の内部に設定する。コード生成部57は、プログラムソースファイル10のコード化を行う。実行ログ採取関数呼び出し判定部58は、コード生成部57により生成されたプログラム目的コードが実行ログ採取関数を挿入すべき箇所（位置）であるかを判定する。実行ログ採取関数呼び出し命令生成部59は、実行ログ採取関数呼び出し判定部58により指定されたプログラム目的コードの箇所に挿入する実行ログ採取関数を呼び出すコードを生成する。

【0019】本実施例の特徴的なことは、プログラムの実行の際にログを出力させ、そのログを採取するための制御情報である実行ログ制御情報をプログラムソースファイル10とは別に実行ログ制御情報ファイル4に記述し、その実行ログ制御情報ファイル4をコンパイル時に読み込むようにしたことであり、この結果、プログラムソースファイル10を修正することなく実行ログを採取することができる。

【0020】図2は、コンパイラ52の内部処理の手順

を示したフローチャートであり、以下、図1、2を用いて、本実施例におけるコンパイル方式の処理手順を説明する。

【0021】コンパイルが開始されると、ステップ101では、オプション判定部52において、コンパイラ52の起動時に指定されたオプションの解析し、指定されたオプションに実行ログの採取を指定する実行ログ採取オプションが指定されているかどうかを判定する。

【0022】ステップ102において、実行ログ採取オプションが指定されている場合、そのオプションに指定された実行ログ制御情報ファイル4から実行ログ制御情報を読み込む。つまり、コンパイラ52の起動時には予め実行ログ制御情報ファイル4を作成しておく必要がある。本実施例における実行ログ制御情報には、実行ログ採取関数名、実行ログ採取関数の引数、ソースプログラムの実行ログ採取関数の挿入する箇所等の実行ログ採取に関するデータが記述されている。このように、実行ログ制御情報を実行ログ制御情報ファイル4に記述しておけば、実行ログ制御情報ファイル4を繰り返し使用でき、また、他にも流用でき、更に複数の実行ログ採取関数を指定することが容易にできる。

【0023】ステップ103では、実行ログ制御情報ファイル解析部55において、実行ログ制御情報ファイル4から実行ログ制御情報の読み込み後、実行ログ制御情報を解析し、所定の形式でコンパイラ52内部の実行ログ情報格納領域に設定する。

【0024】コンパイラ52は実行ログ採取に関するデータが設定されると、プログラムソースファイル10を読み込み、通常の字句解析処理（ステップ104）及び構文解析処理（ステップ105）を行う。

【0025】ステップ106以降の処理で目的コードファイル30を生成するわけであるが、まず、ステップ106において、実行ログ採取オプションが指定されているかを判定する。次に、実行ログ採取オプションが指定されている場合、現在処理対象となっているソースプログラム行が処理され、出力されようとしているプログラム目的コードが実行ログ採取指定された位置のプログラム目的コードかどうか判定する（ステップ107）。もし実行ログの採取が指定された位置ならば、実行ログ採取関数呼び出しコードを生成する（ステップ108）。その後、通常のソースプログラムのプログラム目的コードを生成し、実行ログ採取関数呼び出しコードと併せて目的コードファイル3を生成する（ステップ109）。

【0026】以上のように、本実施例によれば、実行ログ制御情報が記述された実行ログ制御情報ファイル4を用いることでソースプログラムを修正することなく実行ログを採取することができる。

【0027】なお、上記コンパイル方式を備えたコンパイラ52において、ソースプログラムの主関数をコンパイルする時は、さらに実行ログ情報格納領域の初期化処

理と、ソースプログラム本体の処理が終了した時に採取された実行ログ情報をファイルに書き出す処理を行うためのコード生成処理と、ソースプログラム本体が処理の途中でアボートしてもそれまでに採取した実行ログ情報を出力できるような前処理用のコード生成処理を自動的に行う。あるいは、それらの処理をプログラムスタート関数として用意し、プログラム本体とリンクすることによって、実行ログ情報を採取するようにしてもよい。

【0028】また、実行ログ採取関数は、その関数の外部の関数内で定義されている局所変数の値や広域変数の値を実行ログ情報として採取するだけでなく、実行ログ採取関数が呼ばれた時点でのスタック等システム関連の変数値も採取できるように用意しておけば、システム内部状態の変遷を把握することができる。

【0029】また、上記実施例で生成された実行ログ採取関数呼び出しコードでは、その実行ログ採取関数の引数として呼び出した箇所のアドレスがセットされる。したがって通常のデバッグオプションと併用することにより実行ログ採取関数内部で、実行ログ関数を呼び出した関数名、ファイル名、行番号等のシンボル情報を参照することが可能になる。また、実行ログ採取関数自身の処理内容に関してコンパイラ52は認識しなくても良いため、ユーザが必要とする実行ログ情報のみをユーザの独自に設計したデータ構造で出力することができる。

【0030】実施例2. 上記実施例1においては、実行ログ採取オプションに実行ログ制御情報ファイル4を指定し、実行ログ制御情報ファイル4から実行ログ制御情報を読み込むことでソースプログラムを修正することなく実行ログを採取することができるようにした。本実施例においては、実行ログ採取オプションに直接実行ログ情報採取関数名等実行ログ制御情報を指定することを特徴とする。

【0031】これにより、実行ログ制御情報ファイル4をあえて作成しなくても実施例1と同様の効果を得ることができる。少数の実行ログ情報採取関数を指定する場合には特に有効である。この場合、図3におけるステップ102の処理はなく、ステップ103において、実行ログ制御情報は、実行ログ制御情報ファイル4から読み込むのではなく、実行ログ採取オプションから得られることになる。その他の処理は実施例1とほぼ同様である。

【0032】実施例3. 上記実施例1においては、実行ログ制御情報ファイル4に実行ログ制御情報を記述し、この実行ログ制御情報を読み込むことでソースプログラムを修正することなく実行ログを採取することができるようにした。本実施例においては、実行ログ制御情報ファイル4に1ないし複数の実行ログ制御情報ファイル名を記述することを特徴とする。この記述された実行ログ制御情報ファイルには、実行ログを採取するファイル名と実施例1と同様の実行ログ制御情報とが記述されてい

る。

【0033】これにより、実行ログの内容あるいは実行ログ情報採取関数の種類等により別々のファイルに実行ログを採取したい場合等に有効である。この場合、図3におけるステップ102では、実行ログ制御情報ファイル4の内容を読み込む際、実際の実行ログ制御情報であるかファイル名であるかを判定する必要がある。前者の場合は実施例1に相当する。後者の場合は実行ログ制御情報ファイル4に記述されたファイルの内容を読み込み、その後は実施例1と同様の処理を指定ファイル数文繰り返して行えばよい。また、上記の構成によれば、ファイルの指定を階層的にすることは可能である。その他の処理は実施例1とほぼ同様である。

【0034】

【発明の効果】以上のように、この発明によればプログラムソースファイルを修正することなしに、ソースファイル中の任意の箇所に任意の実行ログ採取関数を挿入することができるのでユーザの必要とするプログラムの実行ログ情報を効率良く採取することが可能である。

【0035】実行ログ制御情報をファイルに記述することにより、そのファイルを繰り返し使用することができ、多くの実行ログ採取関数を指定するときにより有効となる。

【0036】また、実行ログ制御情報を実行ログ採取オプションに指定することもでき、このようにすれば、実行ログ制御情報ファイルを作成しなくても上記効果を得

ることができる。

【0037】更に、実行ログ制御情報ファイルに実行ログ制御情報を記述したファイル名を記述することにより、実行ログの内容あるいは実行ログ情報採取関数の種類等により別々のファイルに実行ログを採取することが可能となる。

【図面の簡単な説明】

【図1】この発明におけるコンパイル方式を使用したコンパイラのブロック図であり、同時に処理、データの流れも示した図である。

【図2】この発明におけるコンパイル方式を使用したコンパイラの内部処理手順を示すフローチャートである。

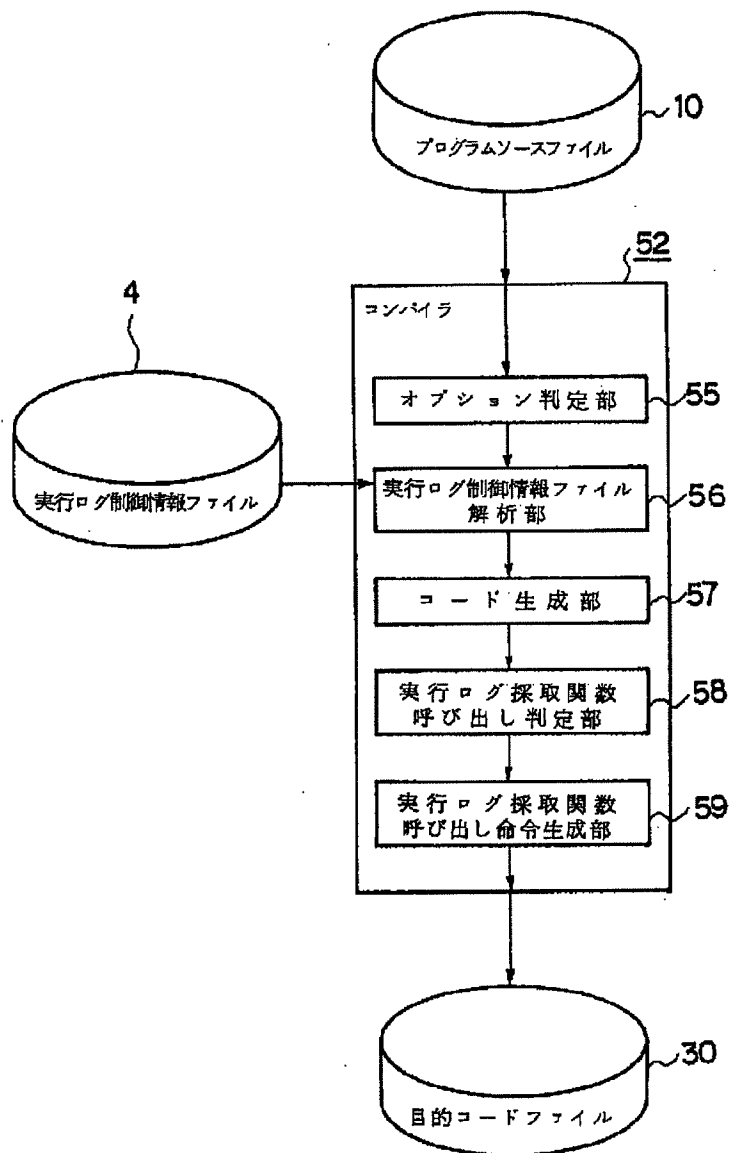
【図3】従来のコンパイル方式を使用したコンパイラのブロック図であり、同時に処理、データの流れも示した図である。

【符号の説明】

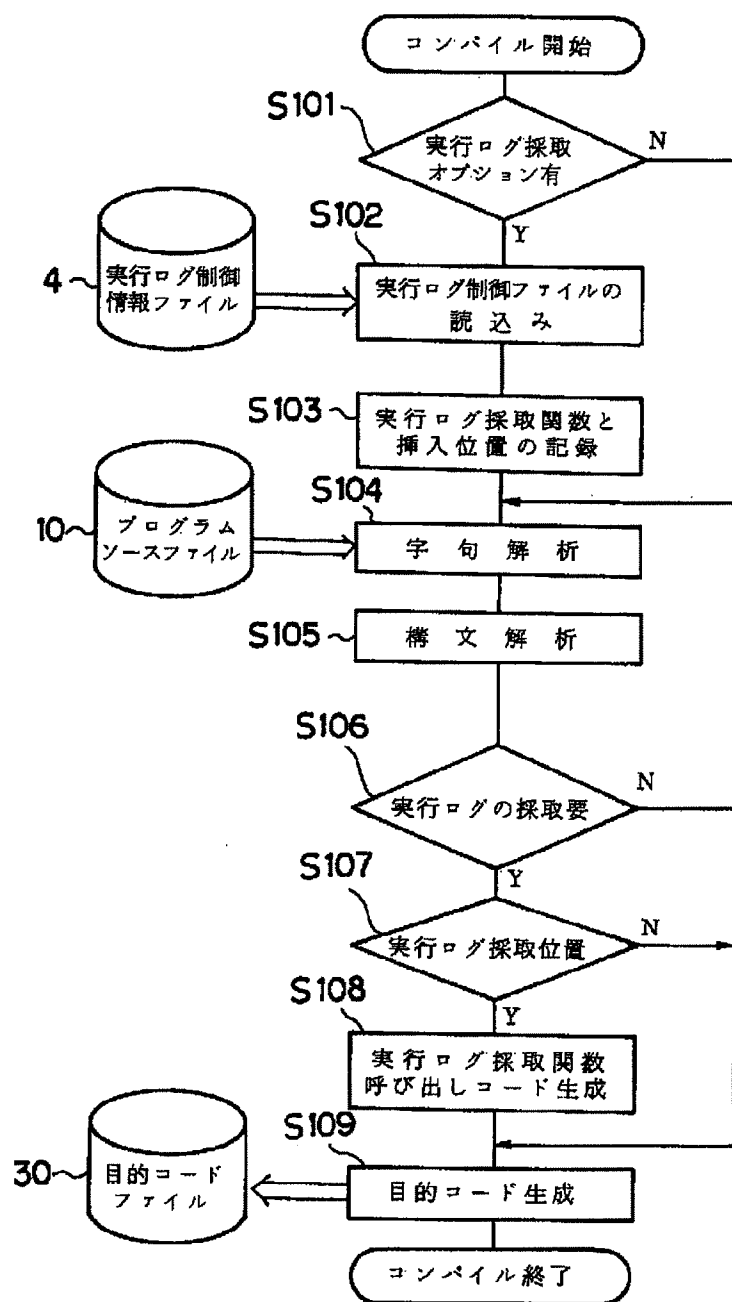
- 1、10 プログラムソースファイル
- 2、52 コンパイラ
- 3、30 目的コードファイル
- 4 実行ログ制御情報ファイル
- 5、55 オプション判定部
- 56 実行ログ制御ファイル解析部
- 7、57 コード生成部
- 8、58 実行ログ採取関数呼び出し判定部
- 9、59 実行ログ採取関数呼び出し命令生成部

【図1】

本実施例におけるコンパイラの内部処理



【図2】



【図3】

従来の実行ログ情報採取のためのコンパイラの内部処理

